面向大数据复杂应用的虚拟集群动态部署模型 *

王　瑾[1,2]，曹云鹏[1,2]，王海峰[1,2†]

(1. 临沂大学 信息科学与工程学院，山东 临沂 276002；2. 山东省网络重点实验室临沂大学研究所，山东 临沂 276002)

摘　要：在大数据复杂应用中会出现多种计算模式混合的作业，因此虚拟集群需要维持多种计算模式的形态进行计算。针对计算负载的时变性和复杂性导致虚拟集群的资源利用率不高的问题，为提高虚拟集群资源的全局利用率，采用弹性资源管理策略来吸收多种计算模式混杂时的资源需求突变。在 Docker 容器技术的支持下提出一个根据作业需求变化的动态部署模型。该模型根据资源的动态需求变化，实时调整虚拟集群的计算形态，具体包括计算节点的类型及规模。该模型不仅实现用户作业执行环境的动态定制，而且达到错峰计算的目的。仿真实验表明，该模型使得虚拟节点 CPU 利用率提升 5.3%，并且优化了计算作业的执行效率。该动态部署模型适合应用到数据中心或大规模集群中，能够有效提高计算资源的利用率。

关键词：虚拟集群；动态部署；大数据复杂应用；Docker 容器

中图分类号：TP393　　doi: 10.19734/j.issn.1001-3695.2018.11.0869

Virtual cluster dynamic deployment model for complex applications in large-scale data processing

Jin Wang[1, 2], Cao Yunpeng[1, 2], Wang Haifeng[1, 2†]

(1. *School of Information Science & Engineering*, *Linyi University*, *Linyi Shandong* 276002, *China*; 2. *Linda Institute*, *Shandong Provincial Key Laboratory of Network Based IntelligentComputing*, *Linyi Shandong* 276002, *China*)

Abstract: There are an amount of computing jobs with multiple computing modes in complex application scenarios. The virtual cluster needs to maintain different computing modes for the large-scale data computing jobs. To deal with the low resource utilization of virtual cluster due to the time-varying and complexity of workloads, this paper used the flexible resource management strategy to improve the global resource utilization of virtual cluster and absorb the mutation of resource demand when multiple computing modes are mixed. This paper proposed a novel dynamic deployment model based on Docker. According to the dynamic demand of workloads, this model can change the computing form of virtual cluster in real time. Its computing form includes the node type and cluster size. This dynamic deployment model not only realizes the dynamic customization of job execution environment, but also achieves the purpose of virtual cluster peak calculation. Simulation experiments show that this model can improve the virtual node CPU utilization by 5.3% and improve the task execution efficiency. This model is suitable for data centers or large-scale clusters to improve the utilization of computing resources.

Key words: virtual cluster; dynamic deployment; large-scale complex applications; Docker container

0　引言

大数据分析和实时查询是一种复杂的计算场景,其中会出现大量的离线分析或者在线查询作业，这类作业可能包括以 MapReduce 为主的多种计算模式(批处理、流计算、内存计算、图计算等)。MapReduce 是谷歌提出的一种经典的分布式并行处理计算范式，分成 Map 任务分发和 Reduce 结果收集两个阶段。MapReduce 的设计初衷是针对批处理计算模式，适合处理离线吞吐量敏感的计算作业。随着应用的发展而扩展出流计算、内存计算、图计算和交互计算等模式。将 MapReduce 的中间结果暂存到内存中则是内存计算，适合处理包含大量迭代的计算作业；流计算需要实时处理连续数据流，将无界数据流划分成固定大小的有界批处理子集；图计算可转换成有依赖关系的多个子任务的 MapReduce 迭代计算形式[1]。总之，大数据分析和实时查询是一个复杂的计算应用场景，包含以 MapReduce 批处理和其扩展形式的多种计算，甚至多种计算模式的组合。另一方面，在处理大数据复杂应用的大规模集群或者云环境中，虚拟技术是解决物理资源利用率低的有效途径。然而虚拟计算环境的资源利用率也仅维持在 10%~17%之间[2]，因此虚拟计算环境的资源利用尚且存在较大的优化空间。

目前轻量级容器 Docker 技术逐渐成熟，开始成为部署虚拟集群和云环境的重要工具[3]。轻量级容器比传统虚拟机的性能损失小且启动速度快[4]，适合在大数据复杂计算场景中实施应用。由于大数据计算是计算资源密集型的作业，所以通过轻量级容器技术能够提高计算资源的利用率和性能。本文针对大数据复杂应用环境中，虚拟集群静态部署方案资源利用率低的问题，提出一种可调整虚拟集群计算形态的动态部署方案。根据计算负载的变化来优化虚拟计算节点的类型和规模，实现作业执行环境动态定制的功能，优化虚拟集群

收稿日期：2018-11-23；**修回日期：**2019-01-23　**基金项目：**山东省自然科学基金面上项目（ZR2017MF050）；山东省高等学校科学技术计划项目（J17KA049）；山东省重点研发项目（2018GGX101005，2017CXGC0701，2016GGX109001）；山东省自主创新及成果转化专项项目（2014ZZCX02702）

作者简介：王瑾（1978-），女，讲师，硕士，主要研究方向为分布式计算；曹云鹏（1967-），男，副教授，硕士，主要研究方向为大数据并行计算；王海峰（1976-），男（通信作者），教授，硕导，博士，主要研究方向为高性能计算、大数据分析（gadfly7@126.com）.

的资源利用率。

1　相关工作

虚拟机的部署是根据用户服务请求将虚拟机部署到合理的物理服务器中，并且为用户提供有效服务[5]。数据中心内的虚拟机部署和迁移一直是研究热点，国内关注理论模型及性能仿真研究。为了提高数据中心物理资源的利用率，从计算资源优化角度分成 CPU 和内存等计算资源、网络带宽资源和能耗资源三个方面。在计算资源优化方面，利用多目标粒子群智能算法来优化数据中心虚拟机部署问题，以提高资源利用率和减少系统响应时间为优化目标，通过线性递减惯性权重法来寻找部署最优解[6]；提出一种动态规划 CPU 和内存资源的虚拟机部署方案，根据未来一段时间内虚拟机需求增量的预测来确定分配规则，仿真结果表明该模型能够有效减少计算资源的分配碎片[7]。在网络带宽资源优化方面，针对数据中心光电网络体系的特点，设计优化子图嵌入算法，通过控制参数来决定两种网络资源的利用比。仿真实验表明该部署优化算法有效降低数据中心带宽分配拒绝率，提高网络带宽利用率[8]。综合考虑虚拟机在 CPU、内存和网络资源的需求，提出感知网络的周期性资源重配置方案。通过适当的虚拟机再调度和迁移，提高数据中心整体的网络通信效率和计算任务的性能[9]。在计算能耗资源研究方面，为了降低大规模数据中心能耗，提出三阈值节能虚拟机部署算法。利用能耗与处理器利用率的线性关系，将负载过重或过轻的虚拟机迁移到负载适度的物理机上。仿真结果表明，在保证服务质量前提下有效降低数据中心能耗[10]。国外的研究者多数从虚拟机部署工具的方面进行研究。在 Docker 上的 EC4Docker 系统为应用程序提供可定制的执行环境，该系统在 Docker Swarm 的基础上弹性管理虚拟节点[3]。轻量级容器编排原型系统有 Roboconf、 SALSA、GRyCAP、Occopus。上述系统属于理论研究型的原型系统，具有各自的编排语言和规则，性能表现也不相同；此外商业级的容器编排系统有 Cloudify、Heat、 CloudFormation、Terraform[11]。

从研究方法的角度而言，通过预测计算负载的变化实现虚拟节点部署是一种有效思路。为了改善弹性云环境部署的高时延，提出一种基于 ARIMA 模型和季节指数的动态负载预测及资源估算方法，利用计算负载与虚拟机配置的关系来估算虚拟机需求量，从而完成虚拟机部署。该方案针对云计算的常规应用场景，采用粗粒度的采样频率来解决细粒度监测方案的不稳定性[12]；为提高对计算负载预测的准确性，设计一种改进 BP 神经网络算法预测物理节点负载，再实施虚拟机加权部署，使虚拟机合理的映射到物理机上。仿真结果表明该方案提高了虚拟机部署的稳定性[13]。研究者已经关注大数据计算负载的研究，Xavier 等人[14]研究 MapReduce 计算的容器级虚拟集群，发现基于轻量级容器的虚拟集群在大数据计算中有较好的性能表现。本文则针对大数据复杂应用场景的计算负载变化，提出一种解决轻量级容器静态部署时资源利用率低的动态部署模型，用于根据计算负载的变化来调整虚拟集群的计算形态。主要贡献点是：基于轻量级容器技术的虚拟集群来处理大数据复杂应用场景，计算负载变化具有复杂性和时变不确定性。动态部署模型提高虚拟节点的资源利用率，并且可实现计算作业执行环境的可定制功能。

2　问题描述

在大数据计算复杂应用场景中，虚拟集群需要预先划分成多个子集，各子虚拟集群执行不同的计算模式。如图 1 所示，在一个虚拟集群中存在批处理、内存计算和流计算三个逻辑上的子集群，在各子集群中执行相应计算模式的任务。对于计算作业的执行存在以下两种情况：a）单个作业要经过多个阶段的不同计算模式才能完成，如图 1 所示，一个作业经过批处理、内存计算和流计算三种计算模式的处理，在各阶段虚拟子集群要保持满负荷状态运行，必然会浪费大量计算资源；b）在多租户场景中处理多个作业时，虚拟集群会出现单个子集群高负载运行的现象，而其他子集群以低负载状态运行，因此要浪费大量虚拟计算资源。

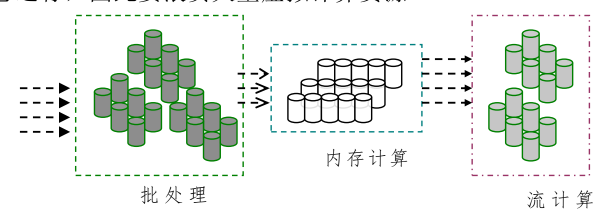


图 1　复杂作业的不同计算阶段的应用场景

Fig. 1　Complex job application scenario with different stages

为了提高虚拟集群的资源利用率，在此应用场景中建立一个虚拟集群动态部署模型。采用轻量级虚拟技术动态生成各种计算模式的子集群，改变虚拟集群的计算形态。计算形态即运行不同计算模式子集群的规模。如图 2 所示，在物理集群基础上创建虚拟集群，为虚拟集群生成特定计算模式（批处理、内存计算、流计算、图计算等）的子集群，执行完毕后释放虚拟节点。

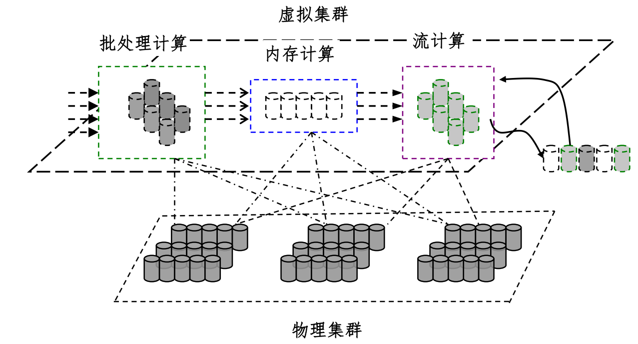


图 2 虚拟集群动态部署示意图

Fig. 2　Virtual cluster dynamic deployment diagram

3　虚拟集群部署模型

3.1　模型的形式化表示

虚拟集群部署模型由一个七元组表示 $DM=<J,\Psi,\mathrm{I},VM,E,R,f>$，其中：$J$ 为大数据作业集合；$\Psi=\{mr,s,m,q\}$ 表示计算模式集合；{ mr 表示 MapReduce 模式，m 表示内存计算，s 表示流计算，q 表示图计算}；I 表示一个作业的计算模式序列；VM 表示虚拟集群，虚拟集群的重要属性是节点类型和规模。例如，一个大数据作业集合 $J=<J_1,J_2,\ldots,J_m>$，计算模式序列矩阵如式 1 所示，对各种主题计算虚拟节点的需求矩阵如式 2 所示。

$$I_J=\begin{bmatrix} mr & s & mr & m\ 0\ldots0 \\ mr & m & s & mr\ 0\ldots0 \\ \ldots & \ldots & \ldots & \ldots\ \ldots\ldots0 \\ mr & s & mr & q\ 0\ldots0 \end{bmatrix} \quad (1)$$

$$VM_J=\begin{bmatrix} 100 & 50 & 20 & 10\ 0\ldots0 \\ 50 & 5 & 60 & 10\ 0\ldots0 \\ \ldots & \ldots & \ldots & \ldots\ \ldots\ldots0 \\ 20 & 5 & 8 & 50\ 0\ldots0 \end{bmatrix} \quad (2)$$

依据式（1）（2），综合两个矩阵的第一行元素 I_{J11},VM_{J11}

可表示作业 J_1 依次使用 100 个 MapReduce 节点，50 个流计算节点，20 个 MapReduce 节点和 10 个内存计算节点才能完成计算。

3.2　作业资源需求模板

为了实现虚拟集群的动态部署模型，先要掌握当前时刻计算作业对资源的需求，本文不详细讨论数据中心计算资源预测问题。假设待处理作业集的类型已知而且数据规模不确定，已知类型作业的资源需求可预测、可量化。解决计算作业资源需求的思路是利用历史数据实现作业分类，根据作业类型和数据规模来预测未来作业的资源需求量，然后实时调整虚拟集群的部署方案。

为了实现计算作业的类型预测和资源需求量化，采用作业资源需求模板的形式化表示方法。一个大数据计算作业在计算过程中分成多个阶段，资源需求是在各阶段需要的资源类型及规模。例如，一个轨迹大数据的相似性查询运算，要经历批处理计算、流计算、图计算、批处理四个阶段，需要相应节点数量为(10,5,3,12)。计算作业资源需求模板的基本语义要素如表 1 所示。CPU 类型属性可适应不同计算模式的变化，根据该属性来确定不同计算模式的虚拟子集群；CPU、内存和输入数据规模用细粒度量化方式，而磁盘、网络带宽、优先级等数据用粗粒度的等级值。在作业资源模板上通过资源需求参数来量化作业的需求信息。用户提交作业时填写作业基本信息，再由资源调度来预测作业信息，并且形成作业的资源需求模板，计算作业的资源需求模板采用 XML 文件来表示。

表 1　计算作业资源需求模板

Table 1　Requirement template of computing jobs

资源属性	属性参数	注释
CPU_Type	CPU==*mr*||*m*||*s*||*q*	CPU 核的计算模型
Resource_Req	CPU==1 && Memory==2GB	对 CPU 和内存的需求
Disk_IO	medium	磁盘的需求，分成 3 个等级
NetWork	High	网络的需求，分成 3 个等级
Input_Data Size	200GB	输入数据的实际规模
Priority	low	优先级分高和低两个等级

3.3　动态部署机制

采用部署事件触发部署规则的机制来实现动态部署。触发事件集 $E=\{e_1,e_2,\ldots,e_i\}$ 包括集群性能和环境事件两种类型。部署规则集 $R=\{r_1,r_2,\ldots,r_m\}$ 中蕴涵专家知识的规则，该规则实现虚拟节点的增加、删除、替换和更新操作。虚拟集群的动态部署机制是触发事件集与部署规则集的函数映射，如式(3）所示。

$$f:\{E\rightarrow R\} \tag{3}$$

设第 i 时刻计算作业的资源需求为 Rq_i，虚拟集群为 $\mathrm{VC}_i=\{\mathrm{VC}^{mr}_i,\mathrm{VC}^{m}_i,\mathrm{VC}^{s}_i,\mathrm{VC}^{q}_i\}$，其中虚拟集群根据批处理计算、内存计算、流计算和图计算分成四个不同的计算主题。触发事件为 $\mathrm{E}_i=\{e_1,e_2,\ldots,e_k\}$，部署规则为 $\mathrm{R}_i=\{r_1,r_2,\ldots,r_m\}$，则第 $i+1$ 个时刻后经过动态部署执行虚拟集群为 $\mathrm{VC}_{i+1}=\{\mathrm{VC}^{mr}_{i+1},\mathrm{VC}^{m}_{i+1},\mathrm{VC}^{s}_{i+1},\mathrm{VC}^{q}_{i+1}\}$。

3.3.1 触发事件

触发事件是启动虚拟集群重新优化部署的先决条件。触发事件由底层信号和相应阈值构成。信号分为计算资源监控和作业资源需求信号两部分。计算资源监控信号有 CPU、内存、磁盘和网络利用率。作业资源需求信号是通过解析某一时刻所有作业需求模板的数据获得，包括对各种计算模式处理器的需求量、内存、磁盘和网络资源的需求。针对每种信号通过人工经验来设置相应阈值，以此产生触发事件，如表 2 所示。

表 2　触发事件

Table 2　Trigger event

事件	事件内容	注释
e_1	Resource_req > ε_1	当前资源需求超过上限阈值时增加节点；
e_2	Resource_req < δ_1	当前资源需求低于下限阈值时减少节点；
e_3	$\sum_{i=1}^{4} CPU^i = \sum_{i=1}^{4} CPU_\mathrm{Re}q^i$	当四种不同主题集群的 CPU 数量和当前 CPU 数量相等时更新各主题集群的节点规模和配置；
e_4	Resource 利用率< δ_2	当前硬件资源利用率低于下限阈值时挂起节点；
e_5	Resource 利用率> ε_2	当前硬件资源利用率超过上限阈值时唤醒节点；

3.3.2 部署规则

部署规则在逻辑上主要分成虚拟节点状态管理、节点管理和集群管理三部分，如表 3 所示。节点状态管理负责虚拟集群中各节点的状态调整，如创建、启动和关闭等；节点管理负责管理集群中节点的配置和规模；集群管理负责管理各种计算模式子集群的状态和配置。

表 3　部署规则列表

Table 3　Deployment rule

类型	规则	例如	注释
状态管理	启动节点	Start(VM$_i$,…VM$_n$)	启动满足条件的节点
	关闭节点	Close(VM$_i$,…VM$_n$)	关闭满足条件的节点
	挂起节点	Suspend(VM$_i$,…VM$_n$)	挂起满足条件的节点
	回复节点	Restore(VM$_i$,…VM$_n$)	恢复满足条件的节点
	迁移节点	Migrate(VM$_i$,…VM$_n$)	迁移满足条件的节点
配置管理	添加节点	Add(VM$_i$,…VM$_n$)	添加特定配置节点
	删除节点	Del(VM$_i$,…VM$_n$)	删除特定配置节点
	更新节点	Update(VM$_i$,…VM$_n$)	更新特定配置节点
集群管理	加入分组	AddGroup(VM$_i$,…VM$_n$, VC)	将节点加入特定分组
	移除分组	DelGroup(VM$_i$,…VM$_n$, VC)	将节点移除特定分组

3.3.3 部署算法

算法-1　虚拟集群动态部署算法

1. 初始化虚拟集群，为各主题集群部署适量节点 $\{\mathrm{VC}^{mr}_i,\mathrm{VC}^{m}_i,\mathrm{VC}^{s}_i,\mathrm{VC}^{q}_i\}$；
2. 初始化事件队列，添加启动部署事件；
3. While E is not Empty Do
4. Collect Singles from $\{\mathrm{VC}^{mr}_i,\mathrm{VC}^{m}_i,\mathrm{VC}^{s}_i,\mathrm{VC}^{q}_i\}$;
5. 根据监控信号生成事件 e_i，各种事件进入事件队列 E；
6. e = deQuery(E);
7. 检索事件—规则库，生成具体部署规则；
8. 执行部署规则；
9. EndWhile

动态部署算法通过循环遍历当前事件队列，然后根据事件集与规则集的映射生成对应部署规则，再逐一执行部署直到事件队列为空。当有新的事件加入事件队列后继续上述过程。该算法中事件与规则映射集用 XML 文件来存储，可实现用户自定义新的部署规则。

3.4　虚拟动态部署系统设计

虚拟集群动态部署系统结构如图 3 所示。从逻辑上主要分成计算作业需求预测、触发事件和动态部署三部分。在计算作业队列中提取用户作业，根据作业类型预测填写资源需求模板信息，并且传送到动态部署模块中；分布式采集器负责收集虚拟集群运行时的各种状态数据，然后汇总到事件发生器中，结合事件数据库来产生触发事件；动态部署模块在当前作业资源需求数据、触发事件和规则库的支持下，创建

动态部署计划并发送到虚拟集群中去执行，实现虚拟节点的增加、删除、更新来创建一个最优的计算执行环境。

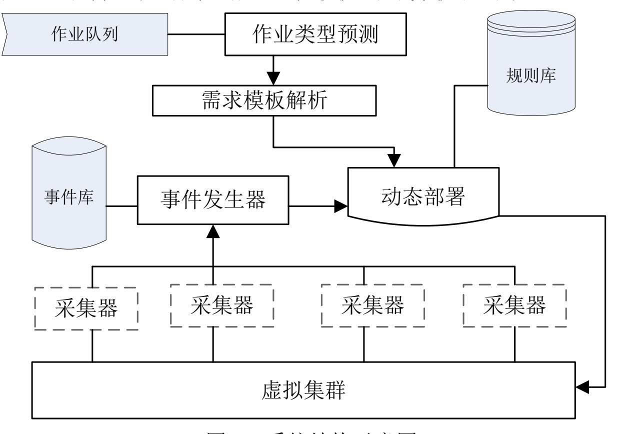


图 3　系统结构示意图

Fig. 3　System architecture diagram

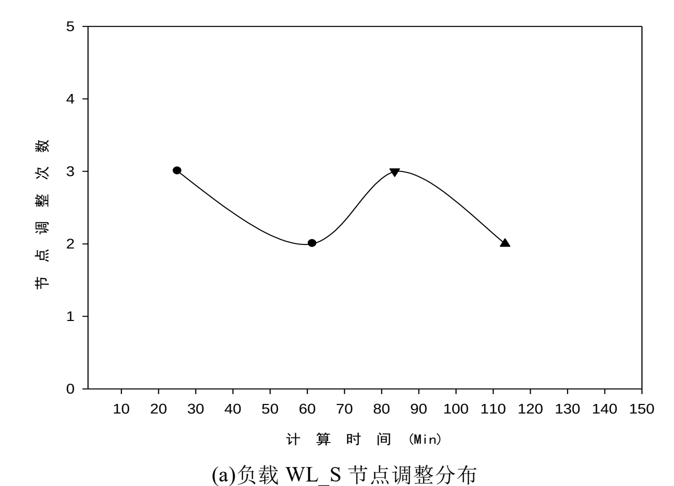
4　实验及分析

实验目标是验证动态部署模型的有效性和资源利用率的优化效果。在实验硬件方面，采用 Intel i7 8700CPU 六核心，32 GB 内存的 64 位机器搭建 16 个节点的物理集群，每个物理机运行 Linux3.10 内核的 Ubuntu Xenial 16.04。每个物理节点上运行 Docker1.7.1 来创建虚拟节点，同时最多运行 3 个虚拟容器，虚拟集群的最大规模为 48 个节点。在 Docker 中用 CPU-Share 比重参数将 6 个 CPU 核心平均分给 3 个虚拟节点，即每个 Container 公平使用 2 个物理 CPU 核心。在实验软件方面，使用 Docker Swarm 实现容器的动态管理，将系统执行规则转换为 Swarm 的管理脚本并且执行。其次，选择谷歌公司的开源系统 CAdvisor 来监控虚拟集群的性能。由于 CAdvisor 智能记录 2 min 以内的性能数据，因此使用支持 CAdvisor 接口的时序数据库 influxDB 保存虚拟集群的性能数据。

在计算负载方面，由于没有针对性的计算负载，所以采用仿真负载来模拟多个计算作业实施实验。由 Hadoop 完成批处理计算负载的任务，Spark 完成内存计算，Spark Streaming 完成流计算，Spark GraphX 完成图计算的仿真负载。预先设置四种计算类型的虚拟节点：Hadoop、Spark、Spark Streaming 和 Spark GraphX 的映像文件，在动态部署时由本地直接加载映像文件完成相应节点的创建。实验与静态部署方案作为基准对比，简记为 SD（static deployment），所提动态部署方案简记为 DM。静态虚拟集群由人工经验来设置各类计算节点比例为 4:3:3:2，即 16 批处理计算节点，12 个内存计算节点，12 个流计算节点和 8 个图计算节点。此外 DM 模型的初始化形态也采用该静态配置方案。

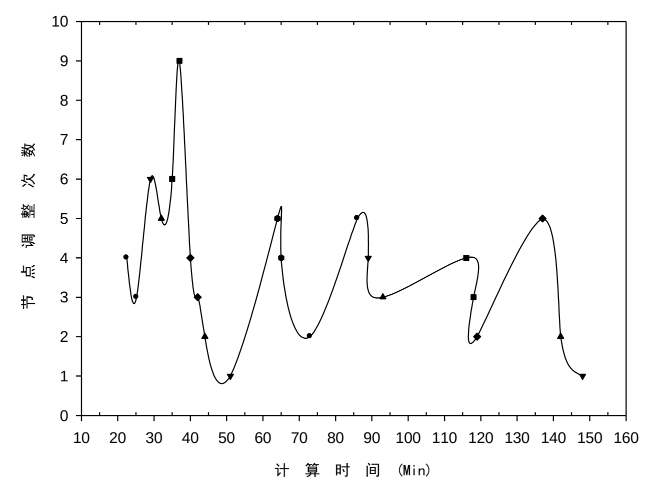
4.1　有效性分析

利用仿真负载来验证动态部署模型的有效性，即能否根据计算负载变化来调整虚拟集群的形态。设计两种大数据计算负载：第一种根据静态部署节点比例而设定的作业类型比例(简记为 WL_S)，其中批计算 16 个、内存计算 12 个、流计算 12 个和图计算 8 个。启动虚拟集群 10 min 以内 48 个作业全部进入调度队列中，在各自节点中运行约 20~30 min，同时并发调度 3 或 4 个作业。第二种仿真负载不按照节点比例设定（简记为 WL_D），其中批计算 12 个、内存计算 18 个、流计算 16 个和图计算 5 个。在仿真负载 WL_D 中降低了批计算和图计算的负载量，增加了内存计算和流计算的负载。



(a)负载 WL_S 节点调整分布

(a) Node adjusting distribution of workloads WL_S



(b)负载 WL_D 节点调整分布

(b) Node adjusting distribution of workloads WL_S

图 4　在不同负载下动态部署模型节点调整情况对比

Fig. 4　Comparison of node adjusting with different workloads

图 4 提供了动态部署模型在两种不同仿真负载运行时，虚拟节点的状态改变次数及对应的计算时间。在负载 WL_S 运行时，出现了四次虚拟节点状态的改变；在负载 WL_D 运行时出现了大量的虚拟节点改变。由于负载 WL_S 是根据初始虚拟节点的比例设定的，虚拟集群不需要调整计算形态就能够较好的完成，由负载的噪声变化引了四次调整，在模型的可接受范围内。然而 WL_D 由于改变了四种计算模式作业的比例，DM 从开始就出现大量节点调整事件，而且每批作业计算结束后也会发生部分节点的调整事件。图 4(b)表明动态部署模型能够根据计算负载的变化，通过调整节点来改变虚拟集群形态。

4.2　性能分析

在本节对上述实验的性能结果进行分析。由于仿真负载的作业都在运行前期提交，实验只统计了作业集合的平均执行时间和总执行时间。面对第一种仿真负载，动态部署模型由于噪声扰动，发生四次节点状态调整。而节点状态调整造成了部分性能损失，在作业平均执行时间和总执行时间上都有表现。对于第二种仿真负载，由于计算资源与作业资源需求不成比例，DM 调整虚拟集群的计算形态来适应负载的资源需求，计算性能的提升掩盖了节点调整的性能损失。从作业平均执行时间和总执行时间来看，动态部署模型取得较好的计算性能提升，作业平均执行时间优化了 7.8%，作业的总执行时间优化了 6.5%。两种负载下的虚拟集群性能对比如表

4 所示。

表 4　两种负载下的虚拟集群性能对比

Table 4　Performance comparison of virtual cluster with two different workloads

性能指标	WorkLoad_S		WorkLoad_D	
	SD	DM	SD	DM
平均执行时间/s	1530	1540	1698	1564
总执行时间/s	7650	7716	9786	9143

然后从虚拟机资源利用率的角度来对比两种方案，主要采用虚拟节点的 CPU 利用率作为监测指标。该监测指标是所有作业执行完毕后各虚拟节点的 CPU 利用率的平均值。监控软件 CAdvisor 利用 Docker API 提取的容器 CPU 利用率需要规范化处理。CPU 利用率是作业进程占用时间的比率。例如，系统中只有一个进程运行在一个 CPU 核上，在一个采用周期（1 000 ms）内该进程占用 CPU 250 ms，则本采用周期 CPU 利用率为 25%；当系统中有 4 个进程运行在 2 个 CPU 核中，每个进程占用 CPU 核心 250 ms，这时采集到的 CPU 利用率为 100%。由于有两个核心，规范处理为 50%。

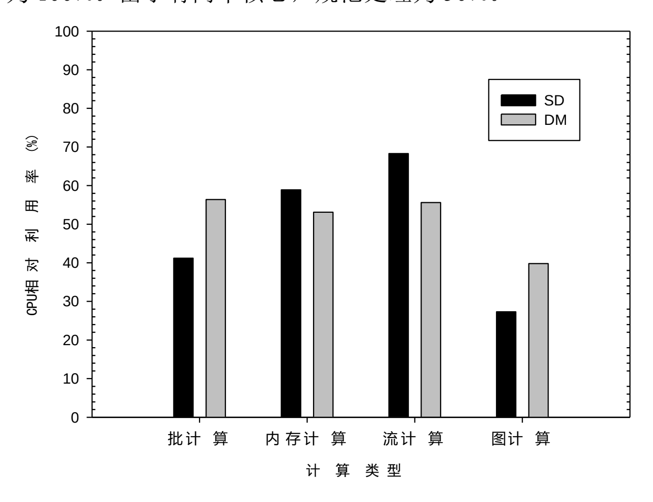


图 5　计算负载 WL_D 各类节点 CPU 利用率对比

Fig. 5　Comparison of CPU utilization rate of workloads WL_D

如图 5 所示，在负载 WL_D 中虚拟集群是减少了批计算与图计算节点的数量，将这片节点转换为内存计算和流计算节点，因此明显提高了批计算和图计算节点的 CPU 利用率，同时降低了内存计算和流计算的负载压力，这点可以从内存计算和流计算节点 CPU 利用率降低体现出来。

5　结束语

虚拟集群的动态配置能够产生更灵活的作业执行环境。轻量级 Docker 容器技术具有较小的启动代价，因此从技术上允许动态部署虚拟机或者实时调整虚拟执行环境。在此研究虚拟集群针对多种模式混合的大数据计算负载，实时调整计算形态的部署策略、方案和系统设计。根据用户提交的作业资源需求和系统的运行信息来触发部署事件，再根据部署规则实施节点调整，最终将部署执行计划转换成 Docker Swarm 的命令脚本。目前动态部署系统需要将各种计算模式节点的 Docker 映像文件本地化存储，并未考虑映像文件远程管理的代价损失。最后仿真负载表明，当负载的资源需求与现有资源配置不匹配时，容器节点的 CPU 利用率能得到有效优化，并且提高了计算作业的执行效率。然而还没有考虑虚拟层资源和物理层资源的均衡问题，以及动态部署稳定性的量化分析。下一步将深入研究动态部署稳定性分析，以及系统扩展性问题，特别是不同网络拓扑下的扩展性研究。

参考文献：

[1] 赵翔，李博，商海川,等. 一种改进的基于 BSP 的大图计算模型 [J]. 计算机学报，2017，40 (1): 223-234. (Zhao Xiang, Li Bo, Sang Haichuan, *et al.* A revised BSP-based massive graph computation model [J]. Chinese Journal of Computers, 2017, 40 (1): 223-234.)

[2] Kaur T, Chana I. Energy efficiency techniques in cloud computing: a survey and Taxonomy [J]. ACM Computing Surveys, 2016, 48 (2): 22-54.

[3] Carlos D A , Amanda C, German M. Container-based virtual elastic clusters [J]. The Journal of Systems and Software, 2017 (127): 1-11.

[4] Zhanibek K, Richard O S. A performance comparison of container-based technologies for the cloud [J]. Future Generation Computer Systems, 2017 (68): 175-182.

[5] 敬超，程小辉. 面向云数据中心的虚拟机部署时延优化算法研究 [J]. 计算机应用研究，2017，34 (12): 3792-3796. (Jing Chao, Cheng Xiaohui. Research of virtual machine placement algorithm for latency optimization on cloud data center [J]. Application Research of Computers, 2017, 34 (12): 3792-3796.)

[6] 黄启成，陈羽中，江伟，等. 一种面向云环境虚拟机部署的粒子群优化策略 [J]. 小型微型计算机系统，2018，39 (7): 1554-1559. (Huang Qicheng, Chen Yuzhong, Jiang Wei, *et al.* A particle swarm optimization strategy for virtual machine deployment in cloud environment [J]. Journal of Chinese Computer Systems, 2018, 39 (7): 1554-1559.)

[7] 张笑燕，王敏讷，杜晓峰. 云计算虚拟机部署方案的研究 [J]. 通信学报，2015，36 (3): 2015084-1-2015084-8. (Zhang Xiaoyan, Wang Minne, Du Xiaofeng. Research on the method of virtual machine deployment in cloud computing [J]. Joumal on Communications, 2015, 36 (3): 2015084-1-2015084-8.)

[8] 邹裕，覃中平. 混合网络的资源分配与虚拟机部署优化算法 [J]. 控制工程，2018，25 (3): 509-515. (Zou Yu, Qin Zhongping. Resource allocation and virtual machine placement optimization algorithm for hybrid networks [J]. Control Engineering of China, 2018, 25 (3): 509-515.)

[9] 罗刚毅，钱柱中，陆桑璐. 一种基于网络感知的虚拟机再调度算法 [J]. 计算机学报，2015, 38 (5): 932-943. (Luo Gangyi, Qian Zhuzhong, Lu Sanglu. A network-aware VM Re-Scheduling algorithm [J]. Chinese Journal of Computers, 2015, 38 (5): 932-943.)

[10] 周舟，胡志刚. 云环境下面向能耗降低的虚拟机部署算法 [J]. 华南理工大学学报：自然科学版，2014，42 (5): 109-114. (Zhou Zhou, Hu Zhigang. A new virtual machine deployment algorithm for energy-consumption reducing in cloud computing [J]. Journal of South China University of Technology: Natural Science Edition, 2014, 42 (5): 109-114.)

[11] Kovacs J. Advances in engineering software [EB/OL]. (2018) [2019-01-21]. https://doi. org/10. 1016/j. advengsoft. 2018. 08. 001.

[12] 文静，李陶深，黄汝维. IaaS 下基于预测的弹性云服务的研究 [J]. 系统工程理论与实践，2014, 34 (SI): 263-268. (Wen Jing, Li Taoshen, Huang Ruwei. Research on elastic cloud service based on prediction under IaaS [J]. Systems Engineering-Theory & Practice, 2014, 34 (SI): 263-268.)

[13] 张蓓蓓，陈宁江，胡丹丹. 基于 BP 神经网络负载预测的虚拟机部署策略 [J]. 华中科技大学学报：自然科学版，2012，40 (SI): 120-123. (Zhang Beibei, Chen Ningjiang, Hu Dandan. A virtual machine deployment strategy based on load prediction of BP neural network [J].

Journal of Huazhong University of Science and Technology: Natural Science Edition, 2012, 40 (S1): 120-123.)

[14] Xavier M G. , Neves M V, De Rose C A F. 2014. A performance comparison of container-based virtualization systems for mapreduce clusters [C]// Proc of the 22nd IEEE Euromicro International Conference on Parallel, Distributed and Network-Based Processing. Italy: Torino, 2014: 299-306.